REMARKS

Claims 1-8 have been canceled without prejudice or disclaimer. New claims 9-15 have been added. Accordingly, claims 9-15 are currently pending in the application.

Priority

Applicants appreciate the Examiner's acknowledgment of the claim for priority and receipt of the certified priority document.

35 U.S.C. §103

Claims 1-8 have been canceled without prejudice or disclaimer in favor of new claims 9-15. It is submitted that new claims 9-15 patentable define the present invention over the cited references.

Accordingly to Applicants invention, a plurality of operating systems (OSs) are executed by a single computer. A program for controlling interrupt requests from hardware devices (such as a keyboard, a display, etc.) that are connected to the computer is provided according to the present system as a device driver (which is referred to as a "multi-OS driver") of the first operating system (OS). Thereafter, when a second OS is activated, address mapping is changed so that

the second OS can also use the multi-OS driver. In order to accomplish this, the multi-OS driver is loaded in the common memory area shared by the various OSs.

The multi-OS driver also manages assignment of all of the hardware devices to the OSs. When an OS wishes to load a new device driver, for adding new hardware device as the OS, the OS asks the multi-OS driver whether or not the OS can use the new hardware device. In response thereto, the multi-OS driver denies the request, if another OS is already using the new hardware device.

On the other hand, when an OS wishes to cease use of a hardware device, the OS notifies the multi-OS driver that the OS has unloaded the device driver corresponding to the hardware device, i.e., terminates the use of the hardware device. In response thereto, the multi-OS driver changes the contents of its own interrupt table in order to make the hardware device corresponding to the unloaded device driver have a state of being usable to all the OSs. When receiving a request for use of the hardware device, from an OS, having the above-mentioned state (i.e., usable to all of the OSs) upon loading of a corresponding device driver by the OS, the multi-OS driver permits the requesting OS to use the hardware device.

In contrast, Solomon discloses a system configuration in which a Unix uses a Software Abstraction Layer (SAL) to execute Windows on a computer executing Unix. However, the Unix manages all hardware devices and therefore no hardware device is assigned only to Windows. Namely, a hardware device is always assigned to Unix. Thereafter, Unix assigns it to Windows. As such, particular software dedicated to the exclusive operation of each hardware device between OSs is not needed in Solomon.

On the other hand, according to Applicants' invention, a plurality of OSs are executed independently of one another. Thus, Applicants' invention employs the above-mentioned multi-OS driver which is neutral to each of the OSs, in order to assign and delete a hardware device dynamically. Solomon fails to disclose or suggest Applicants' multi-OS driver.

The deficiencies in Solomon are not overcome by resort to Kuwatsuru et al. Kuwatsuru et al disclose an environment of multi-OS execution in which an interrupt control program of a first OS is altered so that the interrupt control program of the first OS can call an interrupt program of a second OS directly. However, Kuwatsuru et al do not disclose or suggest Applicants' multi-OS driver which performs assignment and cancellation of a hardware device.

Conclusion

In view of the foregoing amendments and remarks, Applicants contend that the above-identified application is now in condition for allowance. Accordingly, reconsideration and reexamination are respectfully requested.

Respectfully submitted,

Shrinath Malur
Registration No. 34,663
Attorney for Applicant(s)

MATTINGLY, STANGER & MALUR
1800 Diagonal Rd., Suite 370
Alexandria, Virginia 22314
(703) 684-1120
Date: March 1, 2004